# Appeal Brief 12/1/04

USSN: 09/757,427

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**
**BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES**

| | |
|---|---|
| In Re Application of: | Date: February 23, 2004 |
| Ian R. FINLAY et al. | Confirmation No.: 8482 |
| Serial No.: 09/757,427 | Group Art Unit: 2172 |
| Filed: January 10, 2001 | Examiner: Ly, Anh |

For: QUERY EXECUTION IN QUERY PROCESSING SYSTEMS



Honorable Commissioner of Patents and Trademarks
Washington, D.C. 20231

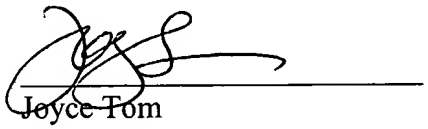
**APPEAL BRIEF TRANSMITTAL LETTER**

Sir:

Submitted herewith are an original and two copies of an Appellant's Brief on Appeal under 37 C.F.R. § 1.192 in connection with the above-referenced Patent application. The Brief includes an Appendix.

Check no. 5791 in the amount of $330.00 is enclosed for payment of the Appeal Brief filing fee. A duplicate of this sheet is attached.

Respectfully submitted,



Joyce Tom
Attorney for Appellants
Reg. No. 48,681
(650) 493-4540

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**
**BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES**

APPEAL NO:

In Re Application of: Ian R. FINLAY et al.

Serial No: 09/757,427

Filed: January 10, 2001

For: QUERY EXECUTION IN QUERY PROCESSING SYSTEMS



# APPELLANT'S BRIEF



Joyce Tom
Attorney for Appellants
International Business Machines
Sawyer Law Group, LLP
2465 E. Bayshore Road, Suite 406
Palo Alto, CA 94303

## III. STATUS OF CLAIMS

Claims 1-18 are pending in the present application and stand rejected. Claims 1, 6 and 11 were amended, and claims 16, 17 and 18 added in a Response filed on April 28, 2003. Accordingly, Claims 1-18 are on appeal and all applied rejections concerning those claims are herein being appealed.

## IV. STATUS OF AMENDMENT

All amendments have been entered.

## V. SUMMARY OF THE INVENTION

The present invention is directed to a method and system for processing a database query on a set of data stored on a plurality of data pages in a database management system. In accordance with the present invention, a query processor calls a data manager to request the return of query-specified data from the set of data. The data manager locates the query-specified data on a stabilized data page and, if appropriate, writes the query-specified data to a buffer while maintaining the stabilization of the data page. The query processor then can retrieve the query-specified data from the buffer.

By writing the query-specified data to a buffer, instead of returning the data directly to the query processor, and maintaining the stabilization of the data page, the data manager operates more efficiently because it can continue to locate query-specified data without repeatedly restabilizing the data page. Query execution is more efficient because the number of times a data page is stabilized is minimized.

The present invention, as recited in claims 1 and 16, provide:

1. A method for processing a database query on a set of data stored on a plurality of data pages in a database management system, the method comprising the steps of:

a) utilizing a query processor to call a data manager and request the return of data from the set of data;

b) allowing the data manager to locate query-specified data on a stabilized data page and make a determination regarding the query-specified data;

c) utilizing the data manager to write the query-specified data on the stabilized data page to a buffer based on the determination while maintaining the stabilization of the data page; and

d) utilizing the query processor to retrieve the query-specified data from the buffer.

16. A method for processing a database query on a set of data stored on a plurality of data pages in a database management system, the method comprising the steps of:

a) utilizing a query processor to call a data manager and request the return of data from the set of data;

b) allowing the data manager to locate query-specified data by:

b1) locating a data page containing query-specified data;

b2) stabilizing the data page; and

b3) accessing the data page;

c) utilizing the data manager to make a determination regarding the query-specified data and to write the query-specified data on the stabilized data page to a buffer based on the determination while maintaining the stabilization of the data page; and

d) utilizing the query processor to retrieve the query-specified data from the buffer.

Independent claims 6 and 11 are claims for a system and a computer readable medium containing program instructions, respectively, that are similar in scope to claim 1. Independent claims 17 and 18 are claims for a system and a computer readable medium containing program instructions, respectively, that are similar in scope to claim 16.

## VI. ISSUES

The issue presented is:

1. Whether claims 1-18 are patentable under 35 U.S.C. § 103(a) over Ponnekanti (U.S. Patent No. 6,363,387).

## VII. GROUPING OF CLAIMS

Appellants hereby state that claims 1-18 do not stand or fall together, but rather claims 1-5 form one group, claims 6-10 for a second group, claims 11-15 for a third group, and claims 16, 17 and 18 each form fourth, fifth and sixth groups respectively.

## VIII. ARGUMENTS

### A. Summary of the Applied Rejections

In the Final Office Action, the Examiner rejected claims 1-18 under 35 U.S.C. §103(a) as being unpatentable over Ponnekanti et al. (U.S. Patent No. 6,363,387). In so doing, the Examiner stated:

> With respect to claim 1, Ponnekanti discloses utilizing a query processor to call a data manager and request the return of data from the set of data (database table storing data pages for query processing as data manager: col. 8, lines 6-20; and query processing calls to access data page table as a set of data: col. 13, lines 8-14); allowing the data manager to locate query-specified data on a stabilized data page (data page storing in the table with indexing: col. 9, lines 1-13 and col. 10, lines 28-32) and make a determination regarding the query-specified data (locating data page and maintaining the data page: col. 8, lines 38-46); utilizing the data manager to write the query-specified data on the stabilized data page to a buffer based on the determination while maintaining the stabilization of the data page (manipulation or operations on index page or data page in the table: col. 16, lines 5-30; also see buffer manager for data page: col. 10, lines 46-67); and utilizing the query processor to retrieve the query-specified data from the buffer

(col. 1, lines 40-50: extracting data page from database table where is storing data page; and col. 10, lines 52-67 and col. 11, lines 1-7).

Ponnekanti does not clearly disclose "to write the query-specified data on the stabilized of data page." However, Ponnekanti discloses the manipulating operations on the data page such as inserting, modifying and deleting (col. 16, lines 5-30, col. 2, lines 11-32 and col. 9, lines 45-60).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to utilize the manipulation operations for data page storing in the table (col. 16, lines 5-30) for reducing overhead as taught by Ponnekanti (col. 3, lines 22-30) because it would have made method for processing data page in the table being optimizing the query processing (col. 4, lines 35-44) and reducing the overhead of locking and increasing the concurrency, a particular performance advantage of the technique in querying data page (col. 10, lines 21-26) and this method would provide the index manager to locate the index and continue scanning the table at the next record row (col. 16, lines 20-40) in the fetching index-data environment.

## B. The Cited Prior Art

Ponnekanti

Ponnekanti is directed to a method for minimizing locking to optimize concurrency. In Ponnekanti, "locking is done after reading the data row, taking into account the status bits of data row and whether the data row qualifies." Col. 4, lines 36-43. During an index scan, Ponnekanti discloses deferred locking where the Index Manager returns RIDs to the Data Layer without acquiring locks on them during scans that have scan arguments on columns not in the index. The Data Layer subsequently qualifies the data row and determines whether locking is really necessary. Thus, the locking is done after the Data Layer reads the data row and determines that the data row qualifies. By determining whether an index or data row qualifies before applying the lock on the row, Ponnekanti reduces locking overhead and increases concurrency.

## B. Independent Claims 1, 6, 11, 16, 17 and 18 Are Allowable Over the Cited Reference.

Appellants respectfully submit that Ponnekanti fails to teach or suggest the present invention, as recited in claims 1, 6, 11, 16, 17 and 18. In particular, Ponnekanti does not teach or suggest "utilizing the data manager to write the query-specified data on the stabilized data page to a buffer . . . while maintaining the stabilization of the data page" and "utilizing the query processor to retrieve the query-specified data from the buffer," as recited in claims 1, 6, 11, 16-18.

In the present invention, during a table scan, the data manager stabilizes, i.e., latches, a data page, locates query-specified data on the stabilized data page, writes the query-specified data from the stabilized data page to a buffer (instead of returning it directly to the query processor) while maintaining the latch on the data page, and continues reading the rows on the data page. The query processor then fetches the data from the buffer. By writing the query-specified data to a buffer, query processing is more efficient because the number of page stabilizations is reduced.

Nothing in Ponnekanti teaches or suggests optimizing query processing *in this manner*. In Ponnekanti, "the action taken by a scan depends on whether the row qualifies and the status of the row status bits. Qualification is done while holding a latch." (Col. 13, lines 15-20). Therefore, qualification entails checking the row status bits and the data. If the row status bits are not set or if the row status bit is "update," and if the data qualifies, the lock manager grants a LOCK on the row and the data manager returns the data row. (Col. 13, lines 23-27; 52-56). Otherwise, the row is either skipped or a LOCK_INSTANT request is processed. (Col. 13, line 28 col. 14, line 2).

Thus, in Ponnekanti, query-specified data on a latched page, i.e., is locked and *returned* to the query processor if the data qualifies and if the row status bit is "unset" or "update."

Ponnekanti makes no mention or suggestion of *writing* "the query-specified data *to a buffer*" after such data has been located and qualified on a stabilized page, nor does Ponnekanti teach or suggest "utilizing the query processor to retrieve the query-specified data *from the buffer*," as recited in claims 1, 6, 11, and 16-18.

In the Final Office Action, the Examiner cites various portions of Ponnekanti which purportedly support his position that Ponnekanti teaches or suggests these features. Appellants respectfully disagree. The cited portions are listed below with a summary of their content.

| **Column, lines cited (in the order presented in the Final Office Action and in the Advisory Action)** | **Summary** |
|---|---|
| Column 16, lines 5-30 | Describes the process for race conditions due to deferred locking. Where an index manager returns the row ID of a qualifying row to a data layer, the data layer must check with the index manager after the data layer latches the data page to determine whether any changes to the index page have occurred. Lines 5-30 describe what the data layer and index manager do in this situation. It does not teach or suggest writing query-specified data to a buffer on a stabilized page. |
| Column 10, lines 46-47 [sic] --- more likely lines 66-67 | Describes generally that latches are held on "kept" pages, which are pages "guaranteed by the database system's buffer manager to not be replaced from the buffer cache." The fact that a buffer cache exists or that kept pages are in the buffer cache does not teach or suggest writing query-specified data to a buffer on a stabilized page. |
| Column 1, lines 40-50 | Discusses database management systems in general. Does not mention or suggest the query processor *retrieving* query-specified data from the buffer. |
| Column 10, line 52 to Column 11, line 7 | Describes latching mechanisms in general. Does not mention or suggest the query processor *retrieving* query-specified data from the buffer. |
| Column 2, lines 11-32 | Discusses database management systems in general and in particular, locking schemes. |
| Column 9, lines 45-60 | Describes generally how a node overflow (in a B-Tree node) is handled by splitting the node. |

Appellants respectfully submit that none of those portions mentions or suggests writing

"the query-specified data on the stabilized data page to a buffer . . . while maintaining the stabilization of the data page" and retrieving "utilizing the query processor to retrieve the query-specified data from the buffer," as recited in claims 1, 6, 11, and 16-18.

In the Advisory Action, the Examiner asserts a somewhat different argument stating that the Access Methods in Figure 2A, item 269 is analogous to the present invention's query processor and that the Lock Manager is analogous to the present invention's data manager. "The Access Methods (as query manager or query processor) calls the Lock Manager (as data manager) to acquire and release locks and latches. Thus, Lock Manager is updating or writing the data page in the buffer based on the query-specified, which is carried out by Access Methods, which is fetching the data page stored in the buffer (col. 10, lines 48-67)."

Appellants respectfully submit that the Examiner's reasoning is unsupported by Ponnekanti. Ponnekanti clearly states that the Access Methods are "*lower-level routines* . . . for carrying out the query-specified operation, such as fetching relevant information (e.g., row 255) from the database table 250." (Col. 7, lines 46-49). Moreover, Ponnekanti explicitly states:

> The Lock Manager is employed for providing interfaces for acquiring and releasing conditional and unconditional locks. . . . The Access Methods (module) calls the Lock Manager/Latch Manager to acquire and release locks and latches. Access Methods, as previously described, provides interfaces functions, such as performing a single table scan or an index scan and inserting a single row, deleting a single row specified by RID, and updating a row specified by RID given its new image, and the like. Access Methods employ an "Index Manager" which manages indices and a "Data Layer" (module) which manages data pages. A "Query Processing Layer" sits on top of Access Methods. It is responsible for splitting complex SQL statements into simpler operations that can be handled by the Access Methods.

Col. 11, line 59 to col. 12, line 8.

According to Ponnekanti, the Access Methods, as opposed to the Lock Manager, is more akin to the data manager of the present invention in that: (1) it is called by the

Execution Unit 268 to fetch query-specified data from a database table, (2) it requests latches for pages, (3) it locates query-specified data on latched pages, (4) requests locks for the rows containing query-specified data, and (5) it returns query-specified data to the Execution Unit 268. Nothing in Ponnekanti teaches or suggests that any of these functions are *performed* by the Lock Manager. Accordingly, Appellants respectfully submit that Ponnekanti's Lock Manager cannot teach or suggest the data manager of the present invention.

Moreover, even if the Lock Manager were to be considered the data manager, which it is not, there is no teaching or suggestion in Ponnekanti that the Lock Manager writes "the query-specified data on the stabilized page to a buffer" and that the Access Methods then retrieves "the query-specified data from the buffer." The portion of Ponnekanti purportedly supporting this assertion only describes locking and latching conceptually. (Col. 10, lines 48-67). It makes no mention of the Lock Manager or of the Access Methods.

Appellants respectfully submit that Ponnekanti fails to teach or suggest the combination of elements recited in 1, 6, 11 and 16-18. Accordingly, claims 1, 6, 11 and 16-18 are allowable. Claims 2-5, 7-10 and 12-15 depend on claims 1, 6 and 11, respectively, and the above arguments apply with equal force. Therefore, Appellants respectfully submit that claims 2-5, 7-10 and 12-15 are also allowable.

**C. Summary of Arguments**

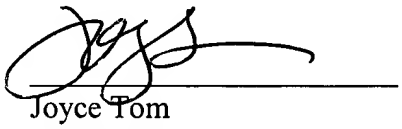
For the reasons set forth above, Appellants respectfully submit that the claims 1-18 are allowable over the cited reference. Appellants respectfully request that the final rejection of

claims 1-18 be reversed.

**Note: For convenience of detachment without disturbing the integrity of the remainder of pages of this Appeal Brief, Appellants' APPENDIX A is attached on separate sheets following the signatory portion of this Appeal Brief.**

This Brief is being submitted in triplicate, and Check no. 5791 in the amount $330.00 is enclosed for payment of the required Brief fee. Please charge any fee that may be necessary for the continued pendency of this application to Deposit Account No. 02-2120 (Sawyer Law Group LLP).

Respectfully submitted,



Joyce Tom
Attorney for Appellants
Reg. No. 48,681
(650) 493-4540

## IX. APPENDIX A

1. (amended) A method for processing a database query on a set of data stored on a plurality of data pages in a database management system, the method comprising the steps of:

a) utilizing a query processor to call a data manager and request the return of data from the set of data;

b) allowing the data manager to locate query-specified data on a stabilized data page and make a determination regarding the query-specified data;

c) utilizing the data manager to write the query-specified data on the stabilized data page to a buffer based on the determination while maintaining the stabilization of the data page; and

d) utilizing the query processor to retrieve the query-specified data from the buffer.

2. (original) The method of claim 1 wherein the determination involves determining whether the query-specified data is to be ignored, consumed, or returned to the query processor.

3. (original) The method of claim 2 wherein the determination is that the query-specified data is to be returned to the query processor.

4. (original) The method of claim 3 wherein the set of data is stored on a plurality of pages and step b) further comprises:

b1) locating a page containing query-specified data;

b2) stabilizing the page; and

b3) accessing the page.

5. (original) The method of claim 4 wherein step c) further comprises:

c1) maintaining the stabilization of the page, while the data manager writes all the query-specified data on the page to the buffer.

6. (amended) A system for processing a database query on a set of data stored on a plurality of data pages in a database management system, the system comprising:

a data manager;

a query processor comprising means to call the data manager and request the return of data from the set of data;

means for allowing the data manager to locate query-specified data on a stabilized data page and make a determination regarding the query-specified data;

means for utilizing the data manager to write the query-specified data on the stabilized data page to a buffer based on the determination while maintaining the stabilization of the data page; and

means for utilizing the query processor to retrieve the query-specified data from the buffer.

7. (original) The system of claim 6 wherein the determination involves determining whether the query-specified data is to be ignored, consumed, or returned to the query processor.

8. (original) The system of claim 7 wherein the determination is that the query-specified data is to be returned to the query processor.

9. (original) The system of claim 8 wherein the set of data is stored on a plurality of pages and the means for allowing the data manager to locate query-specified data further comprises:

means for locating a page containing query-specified data;

means for stabilizing the page; and

means for accessing the page.

10. (original) The system of claim 9 wherein the means for utilizing the data manager to write the query-specified data to a buffer further comprises:

means for maintaining the stabilization of the page, while the data manager writes all the query-specified data on the page to the buffer.

11. (amended) A computer readable medium containing program instructions for processing a database query on a set of data stored on a plurality of data pages in a database management system, the program instructions comprising the steps of:

a) utilizing a query processor to call a data manager and request the return of data from the set of data;

b) allowing the data manager to locate query-specified data on a stabilized data page and make a determination regarding the query-specified data;

c) utilizing the data manager to write the query-specified data on the stabilized data page to a buffer based on the determination while maintaining the stabilization of the data page; and

d) utilizing the query processor to retrieve the query-specified data from the buffer.

12. (original) The computer readable medium of claim 11 wherein the determination involves determining whether the query-specified data is to be ignored, consumed, or returned to the query processor.

13. (original) The computer readable medium of claim 12 wherein the determination is that the query-specified data is to be returned to the query processor.

14. (original) The computer readable medium of claim 13 wherein the set of data is stored on a plurality of pages and step b) further comprises:

b1) locating a page containing query-specified data;

b2) stabilizing the page; and

b3) accessing the page.

15. (original) The computer readable medium of claim 14 wherein step c) further comprises:

c1) maintaining the stabilization of the page, while the data manager writes all the query-specified data on the page to the buffer.

16. (new) A method for processing a database query on a set of data stored on a plurality of data pages in a database management system, the method comprising the steps of:

a) utilizing a query processor to call a data manager and request the return of data from the set of data;

b) allowing the data manager to locate query-specified data by:

b1) locating a data page containing query-specified data;

b2) stabilizing the data page; and

b3) accessing the data page;

c) utilizing the data manager to make a determination regarding the query-specified data and to write the query-specified data on the stabilized data page to a buffer based on the determination while maintaining the stabilization of the data page; and

d) utilizing the query processor to retrieve the query-specified data from the buffer.

17. (new) A system for processing a database query on a set of data stored on a plurality of data pages in a database management system, the system comprising:

a data manager;

a query processor comprising means to call the data manager and request the return of data from the set of data;

means for allowing the data manager to locate query-specified data comprising means for locating a data page containing query-specified data, means for stabilizing the data page, and means for accessing the data page;

means for utilizing the data manager to make a determination regarding the query-specified data and to write the query-specified data on the stabilized data page to a buffer based on the determination while maintaining the stabilization of the data page; and

means for utilizing the query processor to retrieve the query-specified data from the buffer.

18. (new) A computer readable medium containing program instructions for processing a database query on a set of data stored on a plurality of data pages in a database management system, the program instructions comprising the steps of:

a) utilizing a query processor to call a data manager and request the return of data from the set of data;

b) allowing the data manager to locate query-specified data by:

b1) locating a data page containing query-specified data;

b2) stabilizing the data page; and

b3) accessing the data page;

c) utilizing the data manager to make a determination regarding the query-specified data and to write the query-specified data on the stabilized data page to a buffer based on the determination while maintaining the stabilization of the data page; and

d) utilizing the query processor to retrieve the query-specified data from the buffer.